

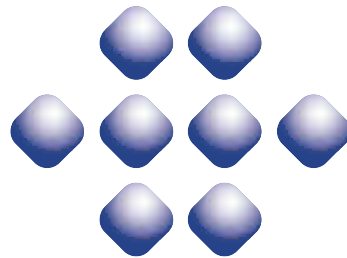
Security Builder® FIPS Module

Version 2.0

FIPS 140-2 Non-Proprietary Security Policy

Certicom Corp.

May 5, 2005



certicom™

©Copyright 2004-2005 Certicom Corp.

This document may be freely reproduced and distributed whole and intact including this Copyright Notice.

“Security Builder” is a registered trademark of Certicom Corp.

Contents

1	Introduction	4
1.1	Overview	4
1.2	Purpose	4
1.3	References	4
1.4	Change Notes	6
2	Cryptographic Module Specification	9
2.1	Physical Specifications	9
2.2	Computer Hardware and OS	10
2.3	Software Specifications	12
3	Cryptographic Module Ports and Interfaces	13
4	Roles, Services, and Authentication	14
4.1	Roles	14
4.2	Services	14
4.3	Operator Authentication	14
5	Finite State Model	17
6	Physical Security	19
7	Operational Environment	20
7.1	UNIX	20
7.2	Linux	21
7.3	Windows	22
8	Cryptographic Key Management	23
8.1	Key Generation	23
8.2	Key Establishment	23
8.3	Key Entry and Output	23
8.4	Key Storage	24
8.5	Zeroization of Keys	24
9	Self-Tests	25
9.1	Power-up Tests	25
9.1.1	Tests upon Power-up	25

9.1.2	On-Demand Self-Tests	25
9.2	Conditional Tests	25
9.3	Failure of Self-Tests	25
10	Design Assurance	26
10.1	Configuration Management	26
10.2	Delivery and Operation	26
10.3	Development	26
10.4	Guidance Documents	27
11	Mitigation of Other Attacks	28
11.1	Timing Attack on RSA	28
11.2	Attack on Biased Private Key of DSA	28
A	Crypto Officer And User Guide	29
A.1	Installation	29
A.1.1	Installing	29
A.1.2	Uninstalling	29
A.2	Commands	29
A.2.1	Initialization	29
A.2.2	De-initialization	29
A.2.3	Self-Tests	29
A.2.4	Show Status	30
A.3	When Module is Disabled	30

1 Introduction

1.1 Overview

This is a non-proprietary Federal Information Processing Standard (FIPS) 140-2 Security Policy for Certicom's **Security Builder[®] FIPS Module Version 2.0** (SB FIPS Module). SB FIPS Module is a cryptographic toolkit for C language users, providing services of various cryptographic algorithms such as hash algorithms, encryption schemes, message authentication, and public key cryptography. This Security Policy specifies the rules under which SB FIPS Module must operate. These security rules are derived from the requirements of FIPS 140-2 [1], and related documents [6, 7, 8].

1.2 Purpose

This Security Policy is created for the following purposes:

1. It is required for FIPS 140-2 validation.
2. To outline SB FIPS Module's conformance to FIPS 140-2 Level 1 Security Requirements.
3. To provide users with how to configure and operate the cryptographic module in order to comply with FIPS 140-2.

1.3 References

References

- [1] NIST *Security Requirements For Cryptographic Modules*, December 3, 2002.
- [2] NIST *Security Requirements For Cryptographic Modules, Annex A: Approved Security Functions for FIPS PUB 140-2*, March 11, 2004.
- [3] NIST *Security Requirements For Cryptographic Modules, Annex B: Approved Protection Profiles for FIPS PUB 140-2*, July 2, 2003.
- [4] NIST *Security Requirements For Cryptographic Modules, Annex C: Approved Random Number Generators for FIPS PUB 140-2*, March 17, 2003.

- [5] NIST *Security Requirements For Cryptographic Modules, Annex D: Approved Key Establishment Techniques for FIPS PUB 140-2*, February 23, 2004.
- [6] NIST *Derived Test Requirements for FIPS 140-2*, Draft, March 24, 2004.
- [7] NIST *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*, July 26, 2004.
- [8] NIST *Frequently Asked Questions for the Cryptographic Module Validation Program*, December 18, 2003.

1.4 Change Notes

The following are placed here by RCS upon check-in.

\$Log: SBFIPSMModule2SecurityPolicy.tex,v \$
Revision 1.26 2005/05/05 14:48:05 ayamada
Revised based on the comments from NIST/CSE.

Revision 1.25 2005/03/28 15:38:18 ayamada
Editorial corrections and refinements.

Revision 1.24 2005/01/28 14:55:32 ayamada
A typo fix.

Revision 1.23 2005/01/19 20:05:59 ayamada
Correction on the date of the document.

Revision 1.22 2005/01/19 20:03:43 ayamada
Added Certificate Number of HMAC.

Revision 1.21 2005/01/19 13:24:19 ayamada
Added OS layer in Figure 2, S/W Block Diagram.
Then, updated the date on the SP.

Revision 1.20 2004/12/01 18:56:37 korr
updated date

Revision 1.19 2004/12/01 16:25:10 korr
updated compatibility matrix

Revision 1.18 2004/10/26 17:36:52 ayamada
1. Certificate numbers are supplied except for HMAC and another ECDSA.
2. Typo fix.

Revision 1.17 2004/10/06 13:34:14 ayamada
Updated the KAT description.

Revision 1.16 2004/08/18 15:53:52 ayamada
Updated for the latest advisory.

Revision 1.15 2004/08/17 16:51:25 ayamada
Added more description to further clarify the FSM.

Revision 1.14 2004/08/17 12:35:40 ayamada
Revised based on the second round comments from the testing laboratory.

Revision 1.13 2004/08/16 19:12:36 ayamada
Minor editorial error correction.

Revision 1.12 2004/08/16 14:39:17 ayamada
Some corrections.

Revision 1.11 2004/08/13 14:15:31 ayamada
Editorial corrections.

Revision 1.10 2004/08/12 17:09:33 ayamada

Major revision based on the first round comments from the testing laboratory.

Revision 1.9 2004/08/09 14:40:14 ayamada
Editorial corrections.

Revision 1.8 2004/08/04 18:00:51 ayamada
A typo fix.

Revision 1.7 2004/08/04 18:00:27 ayamada
Correction in Section 2.1.

Revision 1.6 2004/08/04 17:41:05 ayamada
More information about elliptic curves.

Revision 1.5 2004/08/03 15:09:27 ayamada
Nomenclature update.

Revision 1.4 2004/07/30 16:39:45 ayamada
Figure 2 is added.

Revision 1.3 2004/07/30 15:26:39 ayamada
Figure 2 is added.

Revision 1.2 2004/07/30 12:43:24 ayamada
Further editorial revision.

Revision 1.1 2004/07/30 12:04:39 ayamada
Initial revision: Filename change.

Revision 1.12 2004/07/30 12:00:09 ayamada
Name Change: SB FIPS Provider -> SB FIPS Module

Revision 1.11 2004/07/29 18:10:26 ayamada
Editorial revision.

Revision 1.10 2004/07/14 18:30:42 ayamada
More editorial corrections.

Revision 1.9 2004/07/13 15:38:14 ayamada
Added mitigations of attacks.

Revision 1.8 2004/07/13 13:36:47 ayamada
Editorial corrections.

Revision 1.7 2004/07/09 19:11:09 ayamada
Editorial corrections.

Revision 1.6 2004/07/09 16:05:26 ayamada
Tables 4 and 5 are merged.

Revision 1.5 2004/07/09 14:32:13 ayamada
1. Moved important tables into the main body.
2. Further re-organization.

Revision 1.4 2004/07/07 15:21:41 ayamada
Reorganization of the sections and correct table number references.

Revision 1.3 2004/07/06 20:08:09 ayamada
First editorial revision.

Revision 1.2 2004/07/05 22:59:43 ayamada
Crypto Officer Guide and User Guide are added.

Revision 1.1 2004/07/05 21:42:33 ayamada
Initial revision.

2 Cryptographic Module Specification

SB FIPS Module is a multiple-chip standalone cryptographic module, consisting of the following components:

- A commercially available general-purpose computer hardware.
- A commercially available Operating System (OS) that runs on the computer hardware.
- SB FIPS Module software that runs on the computer hardware and OS.

2.1 Physical Specifications

The general-computer hardware component consists of the following devices:

1. CPU (Microprocessor)
2. Memory
 - (a) Working memory is located on the RAM containing the following spaces:
 - i. Input/output buffer
 - ii. Plaintext/ciphertext buffer
 - iii. Control bufferKey storage is not deployed in this module.
 - (b) Program memory is also located on RAM.
3. Hard Disk (or disks)
4. Display Controller
5. Keyboard Interface
6. Mouse Interface
7. Network Interface
8. Serial Port
9. Parallel Port

10. Power Supply

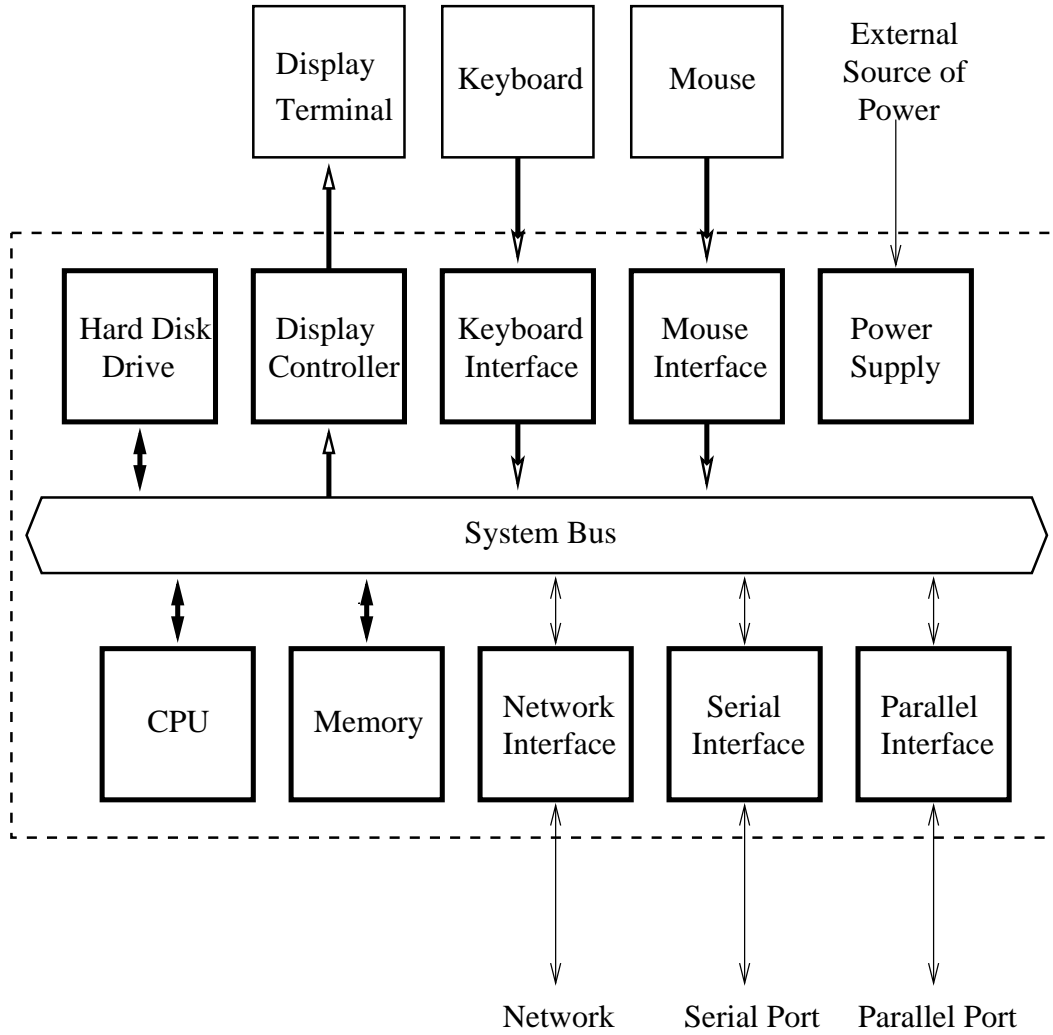
The configuration of this component is illustrated in Figure 1.

2.2 Computer Hardware and OS

The combination of computer hardware and OS include the following representative platforms:

1. Red Hat Linux Application Server 3.0, x86 (Binary compatible to AS 2.1)
2. Solaris 2.9, SPARC 32-bit SPARC
3. Solaris 2.9, SPARC 64-bit SPARC
4. HP-UX 11.00, 32-bit PA-RISC (Binary compatible to 11i)
5. HP-UX 11.00, 64-bit PA-RISC (Binary compatible to 11i)
6. Windows 2003, x86 (Binary compatible to Windows 98/2000/XP)
7. AIX 5.2, 32-bit Power PC (Binary compatible to AIX 5.1)
8. AIX 5.2, 64-bit Power PC (Binary compatible to AIX 5.1)
9. Red Hat Linux Application Server 3.0, Itanium (Binary compatible to AS 2.1)
10. Windows 2003, Itanium
11. HP-UX 11i, Itanium

SB FIPS Module is also suitable for any platforms of any manufactures with compatible processors and equivalent or larger system configurations, and compatible OS versions. For example, an identical SB FIPS Module can be used on any Linux for x86 processors, and an identical SB FIPS Module can be used on all Windows for x86 processors.



[---]: Cryptographic Boundary

↕ : Flow of data, control input, and status output

↓ : Flow of control input ↑ : Flow of status output

Figure 1: Cryptographic Module Hardware Block Diagram

2.3 Software Specifications

The SB FIPS Module component is manufactured by Certicom Corp., providing services to the C computer language users in a shared object format. A single source code base is used for all identified computer hardware and OS.

The interface into the SB FIPS Module shared object is via Application Programmer's Interface (API) function calls. These function calls provide the interface to the cryptographic services, for which the parameters and return codes provide the control input and status output (see Figure 2).

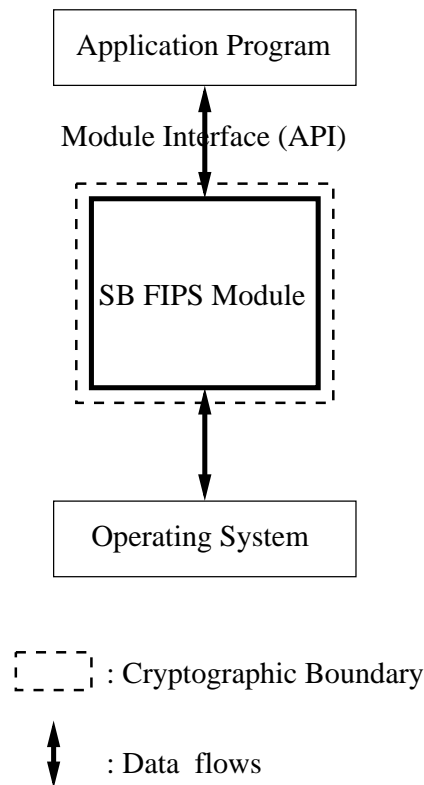


Figure 2: Cryptographic Module Software Block Diagram

3 Cryptographic Module Ports and Interfaces

The physical and logical interfaces are summarized in Table 1.

Table 1: Logical and Physical Interfaces

I/O	Logical Interface	Physical Interface
Data Input	API	Ethernet port
Data Output	API	Ethernet port
Control Input	API	Keyboard and Mouse
Status Output	Return Code	Display
Power Input	Initialization Function (sbg2_FIPS140Initialize())	The Power Supply is the power interface.

4 Roles, Services, and Authentication

4.1 Roles

SB FIPS Module supports Crypto Officer and User Roles, meeting FIPS 140-2 Level 1 requirements. These roles are enforced by this Security Policy. The Crypto Officer has the responsibility for installing SB FIPS Module (see Table 2).

In order to operate the module securely, it is the Crypto Officer and User's responsibility to confine calls to those methods that have been FIPS 140-2 Approved. Thus, in the approved mode of operation, all Roles shall confine themselves to calling FIPS Approved algorithms, as marked in Table 3.

4.2 Services

SB FIPS Module supports many cryptographic algorithms. The set of cryptographic algorithms supported by SB FIPS Module are given in Table 3.

The DES (DES is provided for legacy systems), TDES, AES, SHS (SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512), HMAC-SHS (HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA256, HMAC-SHA-384, and HMAC-SHA-512), RNG, DSA, RSA PKCS #1, and ECDSA algorithms have been validated to comply with FIPS. SB FIPS Module also supports FIPS Approved or allowed key establishment techniques (key agreement and key wrapping), DH, ECDH, ECMQV, and RSA PKCS #1. In order to operate the module in compliance with FIPS, only these FIPS Approved algorithms should be used.

DESX, ARC2, ARC4, MD5, MD2, and HMAC-MD5 are supported as non FIPS Approved algorithms. In order to operate the module in compliance with FIPS, these algorithms should not be used.

4.3 Operator Authentication

SB FIPS Module does not deploy authentication mechanism. The roles of Crypto Officer and User are implicitly selected by the operator.

Table 2: Roles, Services and Access

Service	Crypto Officer	User	Keys and CSPs	Access
Installation, etc.				
Installation	×			
Uninstallation	×			
Self-tests	×	×	ECDSA public key	Use
Show status	×	×		
Symmetric Ciphers				
Key generation	×	×	AES, TDES, DES keys (DES for legacy systems)	Create, Read
Encrypt	×	×	AES, TDES, DES keys (DES for legacy systems)	Use
Decrypt	×	×	AES, TDES, DES keys (DES for legacy systems)	Use
Hash Algorithms and Message Authentication				
Hashing	×	×		
Msg. Authentication	×	×	HMAC keys	Use
Random Number Generation				
Seeding	×	×	Seed	Use
Request	×	×		
Digital Signature				
Key pair generation	×	×	RSA, DSA, ECDSA key pairs	Create, Read
Sign	×	×	RSA, DSA, ECDSA private keys	Use
Verify	×	×	RSA, DSA, ECDSA public keys	Use
Key Agreement				
Key pair generation	×	×	ECDH, ECMQV, DH key pairs	Create, Read
Shared secret generation	×	×	ECDH, ECMQV, DH key pairs	Use
Key Wrapping				
Key pair generation	×	×	RSA key pairs	Create, Read
Wrap	×	×	RSA public keys	Use
Unwrap	×	×	RSA private keys	Use

Table 3: Supported Algorithms and Standards

	Algorithm	FIPS Approved or allowed	Cert Number Number
Block Ciphers	DES (ECB, CBC, CFB64, OFB64) (for legacy systems) [FIPS 46-3]	×	#272
	TDES (ECB, CBC, CFB64, OFB64) [FIPS 46-3]	×	#276
	DESX (ECB, CBC, CFB64, OFB64)		
	AES (ECB, CBC, CFB128, OFB128) [FIPS 197]	×	#175
	ARC2 (ECB, CBC, CFB64, OFB64) [RFC 2268]		
Stream Cipher	ARC4		
Hash Functions	SHA-1 [FIPS 180-2]	×	#260
	SHA-224 [FIPS 180-2]	×	#260
	SHA-256 [FIPS 180-2]	×	#260
	SHA-384 [FIPS 180-2]	×	#260
	SHA-512 [FIPS 180-2]	×	#260
	MD5 [RFC 1321]		
	MD2 [RFC 1115]		
Message Authentication	HMAC-SHA-1 [FIPS 198]	×	#9
	HMAC-SHA-224 [FIPS 198]	×	#9
	HMAC-SHA-256 [FIPS 198]	×	#9
	HMAC-SHA-384 [FIPS 198]	×	#9
	HMAC-SHA-512 [FIPS 198]	×	#9
	HMAC-MD5 [RFC 2104]		
RNG	ANSI X9.62 RNG [ANSI X9.62]	×	#25
Digital Signature	DSS [FIPS 186-2]	×	#115
	ECDSA [FIPS 186-2, ANSI X9.62]	×	#1
	RSA PKCS1-v1.5 [PKCS #1 v2.1]	×	#20
Key Agreement	DH [ANSI X9.42]	×	
	ECDH [ANSI X9.63]	×	
	ECMQV [ANSI X9.63]	×	
Key Wrapping	RSA PKCS1-v1.5 [PKCS #1 v2.1]	×	
	RSA OAEP [PKCS #1 v2.1]	×	

5 Finite State Model

The Finite State model contains the following states:

- Installed/Uninitialized
- Initialized
- Self-Test
- Idle
- Crypto Officer/User
- Error

The following is the important features of the state transition:

1. When the module is installed by the Crypto Officer, the module is in the Installed/Uninitialized state.
2. When the initialization command is applied to the module, i.e., the module is loaded on the memory, turning to the Initialization state. Then, it transits to the Self-Test state automatically, running the Power-up Tests. While in the Self-Test state, all data output via the data output interface is prohibited. On success the module enters Idle; on failure the module enters Error and the module is disabled. From the Error state the Crypto Officer may need to re-install to attempt correction.
3. From the Idle state (which is only entered if self-tests have succeeded), the module can transit to the Crypto Officer/User state when an API function is called.
4. When the API function has completed successfully, the state transits back to Idle.
5. If the Conditional Test (Continuous RNG Test or Pair-wise Consistency Test) fails, the state transits to Error and the module is disabled.
6. When On-demand Self-test is executed, the module enters the Self-Test state. On success the module enters Idle; on failure the module enters Error and the module is disabled.

7. When the de-initialization command is executed, the module goes back to the Installed/Uninitialized state.

6 Physical Security

Physical security is not applicable to this software module at Level 1 Security.

7 Operational Environment

This module is to be run in single user mode on the target operational environment such as UNIX, Linux, and Windows. Therefore the module is restricted to single operator mode operation by the operating system. This section describes the settings for single user mode to be followed.

7.1 UNIX

On UNIX systems, the following operations must be applied [8]:

- Remove all login accounts except “root” (the super user).
- Disable NIS and other name services for users and groups.
- Turn off all remote login, remote command execution, and file transfer daemons.

For example, on HP-UX and Solaris, the following must be applied:

1. Login as the “root” user.
2. Edit the system file `/etc/passwd` and remove all the users except “root” and the pseudo-users. Make sure the password fields for the pseudo-users are a star (*). This prevents login as the pseudo-users.
3. Edit the system file `/etc/nsswitch.conf`. Make sure that “files” is the only option for “passwd” and “group”. This disables NIS and other name services for users and groups.
4. Edit the system file `/etc/inetd.conf`. Remove or comment out the lines for remote login, remote command execution, and file transfer daemons such as `telnetd`, `rlogind`, `remshd`, `rexecd`, `ftpd`, and `tftpd`.
5. Reboot the system for the changes to take effect.

For IBM AIX, the following:

1. Login is as the “root” user.

2. Edit the system file `/etc/passwd` and remove all the users except “root” and the pseudo-users. Make sure that for the pseudo-users, either the password fields are a star (*) or the login shell fields are empty. This prevents login as the pseudo-users.
3. Remove all lines that begin with a plus sign (+) or minus sign (–) from `/etc/passwd` and `/etc/group`. This disables NIS and other name services for users and groups.
4. Edit the system file `/etc/inetd.conf`. Remove or comment out the lines for remote login, remote command execution, and file transfer daemons such as `telnetd`, `rlogind`, `remshd`, `rexecd`, `ftpd`, and `tftpd`.
5. Reboot the system for the changes to take effect.

For DEC OSF1, the following:

1. Login as the “root” user.
2. Edit the system file `/etc/passwd` and remove all the users except “root” and the pseudo-users. Make sure the password fields for the pseudo-users are either a star (*) or “Nologin”. This prevents login as the pseudo-users.
3. Edit the system file `/etc/svc.conf` and make “local” the only option for “passwd” and “group”. Then remove all lines that begin with a plus sign (+) from `/etc/passwd` and `/etc/group`. This disables NIS and other name services for users and groups.
4. Edit the system file `/etc/inetd.conf`. Remove or comment out the lines for remote login, remote command execution, and file transfer daemons such as `telnetd`, `rlogind`, `remshd`, `rexecd`, `ftpd`, and `tftpd`.
5. Reboot the system for the changes to take effect.

7.2 Linux

On Linux systems, the following operations must be applied:

- Remove all login accounts except “root” (the super user).
- Disable NIS and other name services for users and groups.

- Turn off all remote login, remote command execution, and file transfer daemons.

For example, on RedHat Linux, the following must be applied:

1. Login as the “root” user.
2. Edit the system file `/etc/passwd` and `/etc/shadow` and remove all the users except “root” and the pseudo-users. Make sure the password fields in `/etc/shadow` for the pseudo-users are either a star (*) or double exclamation mark (!!). This prevents login as the pseudo-users.
3. Edit the system file `/etc/nsswitch.conf` and make “files” the only option for “passwd”, “group”, and “shadow”. This disables NIS and other name services for users and groups.
4. In the `/etc/inetd.d` directory, edit the files “rexec”, “rlogin”, “rsh”, “rsync”, “telnet”, and “wu-ftp”, and set the value of “disable” to “yes”.
5. Reboot the system for the changes to take effect.

7.3 Windows

In order to ensure single user mode, the following rules:

- All remote guest account must be disabled in order to ensure that up to one human operator can operate SB FIPS Module at a time.
- The use of SB FIPS Module is restricted to a single instance of loading the module to the memory space at a time.
- The actual memory space of the virtual memory space must be on the local machine.

8 Cryptographic Key Management

SB FIPS Module provides the underlying functions to support FIPS 140-2 Level 1 key management. The user will select FIPS Approved algorithms and will handle keys with appropriate care to build up a system that complies with FIPS 140-2. It is the Crypto Officer and User's responsibility to select FIPS 140-2 validated algorithms (see Table 3).

8.1 Key Generation

SB FIPS Module provides FIPS 140-2 compliant key generation. The underlying random number generation uses a FIPS Approved method, the ANSI X9.62 RNG [4].

8.2 Key Establishment

SB FIPS Module provides the following FIPS Approved or allowed key establishment techniques [5]:

1. Diffie-Hellman (DH)
2. EC Diffie-Hellman (ECDH)
3. ECMQV
4. RSA PKCS1-v1_5
5. RSA OAEP

The RSA key wrapping techniques above are based on the PKCS #1 v2.1 standard, and are used to transport keys.

It is users responsibility to ensure that the appropriate key establishment techniques are applied to the appropriate keys.

8.3 Key Entry and Output

It is not allowed to import or export keys, including a seed key, and other security sensitive information outside of physical boundary in plaintext format. For key entry and export, users must ensure to deploy appropriate encryption method using FIPS Approved encryption algorithms in SB FIPS Module or any other FIPS validated cryptographic module.

8.4 Key Storage

SB FIPS Module is a low-level cryptographic toolkit, and as such does not provide key storage.

8.5 Zeroization of Keys

SB FIPS Module functions zeroize all intermediate security sensitive material. All CSPs are zeroized when they are no longer needed by calling destroy functions. Destruction of CSP is enforced in a manner such that missed destruction will make SB FIPS Module no longer functional.

9 Self-Tests

9.1 Power-up Tests

9.1.1 Tests upon Power-up

Self-tests are initiated automatically by the module at start-up. The following tests are applied:

- 1. Known Answer Tests (KATs):**

KATs are performed on DES (DES is provided for legacy systems), TDES, AES, SHS, HMAC-SHS, RNG, and RSA PKCS #1 v1.5 Signature Algorithm. For DSA and ECDSA, Pair-wise Consistency Test is used.

- 2. Software Integrity Test:**

The software integrity test deploys ECDSA signature validation to verify the integrity of the module.

9.1.2 On-Demand Self-Tests

On-demand self tests may be invoked by the Cryptographic Officer or User by invoking a function, which is described in the Crypto Officer And User Guide in Appendix A.

9.2 Conditional Tests

The Continuous RNG Test is executed on all RNG generated data, examining the first 160 bits of each requested random generation for repetition. This ensures that the RNG is not stuck at any constant value.

Also, upon each generation of a RSA, DSA, or ECDSA key pair, the generated key pair is tested of their correctness by generating a signature and verifying the signature on a given message as a Pair-wise Consistency Test.

9.3 Failure of Self-Tests

Failure of the Self-tests places the cryptographic module in the Error state, wherein no cryptographic operations can be performed. If any Self-test fails, the cryptographic module will output error code.

10 Design Assurance

10.1 Configuration Management

A configuration management system for the cryptographic module is employed and has been described in a document to the certifying lab. It uses the Concurrent Versioning System (CVS) to track the configurations.

10.2 Delivery and Operation

Please refer to Section A.1 of Crypto Officer And User Guide in Appendix A to review the steps necessary for the secure installation and initialization of the cryptographic module.

10.3 Development

Detailed design information and procedures have been described in documentation submitted to the certifying laboratory.

This toolkit is designed and developed using high level language C, for C and C++ users. The low level language assembly is used to optimize lower level operations.

Development for the cryptographic module is carried out in a multi-platform environment. Certicom is using the CVS revision control system to control revisions. Development of new versions and major features are performed on a branch of the software, and these branches merged back into the trunk after testing and review, but the branch is maintained to perform post-release maintenance.

Releases are tested first in engineering (via an automated daily procedure, the daily build). They are then committed to the product candidate repository. These releases are then sent to the QA department which must run its own tests before they move them into the product branch. Only QA can move candidates into products.

Daily the software is built automatically on over 45 platforms (servers to embedded devices and PDAs) and the regression tests run. In addition to this, daily integration builds are performed to check the use of the cryptographic library as used in the higher level protocol products.

Weekly, code test coverage and code quality tools (purify and insure) are run on the product (these are very extensive tests taking longer than a day to complete).

Weekly benchmarks are also automatically performed on a representative subset of devices.

10.4 Guidance Documents

Crypto Officer Guide and User Guide are provided in Appendix A. This appendix outlines the operations for Crypto Officer and User to ensure the security of the module.

11 Mitigation of Other Attacks

SB FIPS Module implements mitigation of the following attacks:

1. Timing attack on RSA
2. Attack on biased private key of DSA

11.1 Timing Attack on RSA

When employing Montgomery computations, timing effects allow an attacker to tell when the base of exponentiation is near the secret modulus. This leaks information concerning the secret modulus.

In order to mitigate this attack, the following is executed: The bases of exponentiation are randomized by a novel technique that requires no inversion to remove (unlike other blinding methods e.g. BSAFE Crypto-C User Manual v 4.2).

Note that Remote Timing Attacks are Practical:

<http://crypto.stanford.edu/~dabo/papers/ssl-timing.pdf>

11.2 Attack on Biased Private Key of DSA

The standards for choosing ephemeral values in El-Gamal type signatures introduce a slight bias. Means to exploit these biases were presented to ANSI by D. Bleichenbacher.

In order to mitigate this attack, the following is executed: The bias in the RNG is reduced to levels which are far below the Bleichenbacher attack threshold.

Change Notice 1 of FIPS 186-2 is published to mitigate this attack:

<http://csrc.nist.gov/CryptoToolkit/tkdigsigs.html>

A Crypto Officer And User Guide

A.1 Installation

In order to carry out a secure installation of SB FIPS Module, the Crypto Officer must follow the procedure described in this section.

A.1.1 Installing

The Crypto Officer is responsible for the installation of SB FIPS Module. Only the Crypto Officer is allowed to install the product. The Crypto Officer must have administrative privileges on the computer.

Place the shared object, `libsbgse2.[so,dll]`, in an appropriate location on the computer hardware for your development environment.

A.1.2 Uninstalling

Remove the shared object, `libsbgse2.[so,dll]`, from the computer hardware.

A.2 Commands

A.2.1 Initialization

```
sbg2_FIPS140Initialize()
```

This function runs a series of self-tests on the module. These tests examine the integrity of the shared object, and the correct operation of the cryptographic algorithms. If these tests are successful, a value of `SB_SUCCESS` will be returned and the module will be enabled.

A.2.2 De-initialization

```
sbg2_FIPS140Deinitialize()
```

This function de-initializes the module.

A.2.3 Self-Tests

```
sbg2_FIPS140RunTest()
```

This function runs a series of self-tests, and return `SB_SUCCESS` if the tests are successful. These tests examine the integrity of the shared object, and the correct operation of the cryptographic algorithms. If these tests fail, the module will be disabled. Section A.3 of this document describes how to recover from the disabled state.

A.2.4 Show Status

`sbg2_FIPS140GetState()`

This function will return the current state of the module. For more details, refer to `sbg2fsm.h`.

A.3 When Module is Disabled

When SB FIPS Module becomes disabled, attempt to bring the module back to the Installed state by calling `sb_FIPS140Deinitialize()`, and then to initialize the module using `sb_FIPS140Initialize()`. If the initialization is successful, the module is recovered. If this attempt fails, uninstall the module and re-install it. If the module is initialized successfully by this re-installation, the recovery is successful. If this recovery attempt fails, it indicates a fatal error. Please contact Certicom Support immediately.